Stochastic dual dynamic programming with stagewise dependent objective uncertainty

Anthony Downward[a,*], Oscar Dowson[a], Regan Baucke[a]

[a] *Department of Engineering Science, University of Auckland, New Zealand*

Abstract

We present a new algorithm for solving linear multistage stochastic programming problems with objective function coefficients modeled as a stochastic process. This algorithm overcomes the difficulties of existing methods which require discretization. Using an argument based on the finiteness of the set of possible cuts, we prove that the algorithm converges almost surely. Finally, we demonstrate the practical application of the algorithm on a hydro-bidding example with the spot-price modeled as an auto-regressive process.

Keywords: dynamic programming, decomposition, multistage, SDDP, stochastic programming

1. Introduction

In this paper we consider a multistage stochastic programming problem with stagewise dependent stochastic processes in both the right-hand sides and the objective function coefficients of the stage problem. Multistage convex stochastic programming problems are typically solved using a variant of the SDDP algorithm (stochastic dual dynamic programming) (Pereira and Pinto, 1991). The simplest implementation of this has stagewise independent stochastic processes. The algorithm employs a forward pass, where a sample path is generated from the stochastic process, and a backward pass which refines the approximations of the future cost functions along the sample path from the forward pass.

Stagewise dependent noise is typically modelled by creating additional state variables to keep track of previous noise outcomes. If the noise is on the right-hand

*Corresponding author

Email address: a.downward@auckland.ac.nz (Anthony Downward)

Preprint submitted to Operations Research Letters *July 11, 2018*

sides of constraints, this still results in a value function that is convex (with respect to the state variables). However, if the noise is in the objective, then additional state variables that keep track of this noise will show up in the objective function of the next stage; this leads to a concave value function with respect to these state variables. As a result, the value function is a saddle function. For this reason, standard cutting plane algorithms cannot give valid lower bounds, precluding the modelling of stagewise dependence within the objective coefficients.

Typically, this is partially overcome by modelling the dependent process as a Markov chain with a set of discrete states (Gjelsvik et al., 1999; Philpott and De Matos, 2012; Rebennack, 2016). However, this method can become computationally expensive since cuts must be added for all of the discretized Markov states. Moreover, if the process is multi-dimensional, the number of Markov states required to give a sufficient level of accuracy becomes prohibitive. In stochastic dynamic programming (SDP), a common approach to overcome this restriction is to use an interpolation scheme to evaluate the value function at points in the state space that are not in the discretized lattice. In the SDDP setting, this corresponds to interpolating between the value functions of different Markov states (Gjelsvik et al., 2010; Wahid, 2017).

The work of Baucke et al. (2018) presents a new method for solving multistage minimax stochastic programming problems. This work details a procedure for the generation of valid lower (and upper) bound functions for a saddle function. Our method incorporates these bounding functions and is presented in a style of a standard SDDP implementation, enabling the solution of multistage stochastic programming problems with stagewise-dependent noise modelled in the objective function.

This paper is laid out as follows. In Section 2 we provide a formulation for our multistage stochastic program, and show that the cost-to-go is a saddle function. Then in Section 3 we present the method for generating valid lower representations for saddle functions. In Section 4 we propose an algorithm to solve such problems and provide a convergence proof. Next, in Section 5 we present an example with a discussion on the computational performance of the algorithm. Finally in Section 6 we provide concluding remarks.

2. Formulation

We follow the terminology of Philpott and Guan (2008), restricting our attention to multistage stochastic programs with the following properties:

(1) The set of random noise outcomes Ω_t in each stage $t = 2, 3, \ldots, T$ is discrete and finite: $\Omega_t = \{1, \ldots, N_t\}$, with probabilities p_t^ω, $\forall \omega \in \Omega_t$.
(2) The random noise realizations in each stage are independent of any previous realizations.
(3) The optimization problem in each stage has at least one optimal solution for all reachable states.

Note that we also follow their convention of including the control variables in the state space for notational convenience.

The reader should note that we relax the assumption (A1) in Philpott and Guan (2008) so that the random quantities can appear in any part of the optimization problem. Under these assumptions, the multistage stochastic linear program **SP** can be decomposed into a series of stages, each written in the following form:

$$\begin{aligned}\mathbf{SP}_t: \; V_t\left(x_{t-1}, y_{t-1}, \omega_t\right) = \min_{x_t, y_t} \quad & y_t^\top Q_t x_t + \mathbb{E}_{\omega_{t+1} \in \Omega_{t+1}}\left[V_{t+1}\left(x_t, y_t, \omega_{t+1}\right)\right] \\ \text{subject to} \quad & A_t^{\omega_t} x_t + a^{\omega_t} \geq x_{t-1} \\ & y_t = B_t^{\omega_t} y_{t-1} + b^{\omega_t}\end{aligned}$$

where $A_t^{\omega_t}$ and $B_t^{\omega_t}$ are square matrices, Q_t is an appropriately dimensioned matrix, and a^{ω_t} and b^{ω_t} are vectors. Note that x_t and y_t are both state vectors for stage t. Moreover, $\mathbf{SP}_t$ is not a convex optimization problem due to the bilinear term in the objective. However, observe that, given y_{t-1} and ω_t, y_t is uniquely determined.

In the final time period we solve a variant of this problem where the cost-to-go is set to 0:

$$\begin{aligned}\mathbf{SP}_T: \; V_T\left(x_{T-1}, y_{T-1}, \omega_T\right) = \min_{x_T, y_T} \quad & y_T^\top Q_T x_T \\ \text{subject to} \quad & A_T^{\omega_T} x_T + a^{\omega_T} \geq x_{T-1} \\ & y_T = B_T^{\omega_T} y_{T-1} + b^{\omega_T}\end{aligned}$$

We define the expected cost-to-go function at the end of time period t (prior to observing the noise in period $t+1$) as:

$$\mathcal{V}_{t+1}(x_t, y_t) = \mathbb{E}_{\omega_{t+1} \in \Omega_{t+1}}\left[V_{t+1}\left(x_t, y_t, \omega_{t+1}\right)\right]. \tag{1}$$

The following two lemmas are presented in order to show that $\mathcal{V}_{t+1}(x_t, y_t)$ is a saddle function that is convex in x_t and concave in y_t, for all $t \in \{1, \ldots, T-1\}$.

Lemma 1. *$\mathcal{V}_{t+1}\left(x_t, y_t\right)$ is a convex function of x_t for all $t \in \{1, \ldots, T-1\}$.*

Proof. We will prove this by induction. First observe in SP_T that y_T is uniquely determined for a fixed y_{T-1} and ω_T and thus $\mathbf{SP}_T$ can be cast as a linear program by fixing y_T. Since x_{T-1} appears only as a right-hand side vector, it follows that $V_T(x_{T-1}, *, *)$ is a convex function. Moreover, following (1), $\mathcal{V}_T(x_{T-1}, y_{T-1})$ is a convex function in x_{T-1} given a fixed value for y_{T-1}.

Now suppose that $\mathcal{V}_{t+1}(x_t, y_t)$ is a convex function for a fixed y_t. Given this, $\mathbf{SP}_t$ can be cast as a convex optimization problem (with y_t fixed). Since x_{t-1} appears only in the right-hand side of a set of constraints, we know that $V_t(x_{t-1}, *, *)$ is a convex function of x_{t-1}. This implies that $\mathcal{V}_t(x_{t-1}, y_{t-1})$ is convex with respect to x_{t-1} for a fixed value of y_{t-1}. Since $\mathcal{V}_{t+1}(x_t, y_t)$ is convex in x_t for $t = T - 1$, it is convex for all $t \in \{1, \ldots, T - 1\}$, as required. □

Lemma 2. *$\mathcal{V}_{t+1}(x_t, y_t)$, is a concave function of y_t for all $t \in \{1, \ldots, T - 1\}$.*

Proof. We will also prove this by induction. Since $\mathbf{SP}_T$ can be cast as a linear program by fixing $y_T = B_T^{\omega_T} y_{T-1} + b^{\omega_T}$ as a cost coefficient, it follows that $V_T(*, y_{T-1}, *)$ is a concave function of y_{T-1}. Thus $\mathcal{V}_T(x_{T-1}, y_{T-1})$ is a concave function in y_{T-1}, for a fixed x_{T-1}.

Now suppose that $\mathcal{V}_{t+1}(x_t, y_t)$ is a concave function with respect to y_t. We will show that $\mathcal{V}_t(x_{t-1}, y_{t-1})$ is also concave (with respect to y_{t-1}). Consider two arbitrary vectors of y_{t-1}: y_{t-1}^1 and y_{t-1}^2, given ω_t, the corresponding values for y_t in $\mathbf{SP}_t$ are y_t^1 and y_t^2, respectively. Moreover, suppose for a fixed x_{t-1} the optimal values of x_t are x_t^1 and x_t^2, respectively. Therefore:

$$\begin{aligned} V_t\left(x_{t-1}, y_{t-1}^1, \omega_t\right) &= y_t^{1\top} Q_t x_t^1 + \mathcal{V}_{t+1}\left(x_t^1, y_t^1\right), \quad (2) \\ V_t\left(x_{t-1}, y_{t-1}^2, \omega_t\right) &= y_t^{2\top} Q_t x_t^2 + \mathcal{V}_{t+1}\left(x_t^2, y_t^2\right). \quad (3) \end{aligned}$$

Let us define $y_{t-1}^3 = (1-\lambda) y_{t-1}^1 + \lambda y_{t-1}^2$, where $\lambda \in [0, 1]$. Now suppose that the corresponding optimal solution to $\mathbf{SP}_t$ with $y_{t-1} = y_{t-1}^3$ is $x_t = x_t^3$. Therefore:

$$V_t\left(x_{t-1}, y_{t-1}^3, \omega_t\right) = y_t^{3\top} Q_t x_t^3 + \mathcal{V}_{t+1}\left(x_t^3, y_t^3\right).$$

Since the cost-to-go function is concave with respect to its second argument, we have the following inequality:

$$V_t\left(x_{t-1}, y_{t-1}^3, \omega_t\right) \geq (1-\lambda)\left(y_t^{1\top} Q_t x_t^3 + \mathcal{V}_{t+1}\left(x_t^3, y_t^1\right)\right) + \lambda\left(y_t^{2\top} Q_t x_t^3 + \mathcal{V}_{t+1}\left(x_t^3, y_t^2\right)\right) \quad (4)$$

Note that the right-hand side of (4) gives a convex combination of the objective function values of $\mathbf{SP}_t$ for $x_t = x_t^3$ with y_{t-1} equal to y_{t-1}^1 and y_{t-1}^2. Next observe that although x_t^3 is feasible for both problems, it is not necessarily optimal. Replacing x_t^3 in (4) with the corresponding optimal solutions x_t^1 and x_t^2 gives:

$$V_t\left(x_{t-1}, y_{t-1}^3, \omega_t\right) \geq (1-\lambda)\left(y_t^{1\top} Q_t x_t^1 + \mathcal{V}_{t+1}\left(x_t^1, y_t^1\right)\right) + \lambda\left(y_t^{2\top} Q_t x_t^2 + \mathcal{V}_{t+1}\left(x_t^2, y_t^2\right)\right) \quad (5)$$

Now we substitute equations (2) and (3) into inequality (5):

$$V_t\left(x_{t-1}, y_{t-1}^3, \omega_t\right) \geq (1-\lambda) V_t\left(x_{t-1}, y_{t-1}^1, \omega_t\right) + \lambda V_t\left(x_{t-1}, y_{t-1}^2, \omega_t\right).$$

This gives the result that, if $\mathcal{V}_{t+1}(x_t, y_t)$ is concave w.r.t. y_t then $V_t(*, y_{t-1}, *)$ is concave w.r.t. y_{t-1}. Thus from equation (1), $\mathcal{V}_t(x_{t-1}, y_{t-1})$ is also concave w.r.t. y_{t-1}. Since we know that $\mathcal{V}_T(x_{T-1}, y_{T-1})$ is concave w.r.t. y_{T-1}, we have concavity for $\mathcal{V}_{t+1}(x_t, y_t)$ w.r.t. y_t for all $t \in \{1, \ldots, T-1\}$, as required. □

Lemmas 1 and 2 ensure that our cost-to-go function $\mathcal{V}_{t+1}(x_t, y_t)$ is a saddle function. This is different from standard implementations of multistage stochastic programs, where the cost-to-go is convex, allowing a lower-bound function to be approximated by linear cutting planes. We note, however, that since we have no action that influences the value of y_t, we can introduce a new minimax form of the stage problem $\mathbf{SP}_t$, named $\mathbf{SP}'_t$, which we will minimize with respect to x_t and maximize with respect to y_t.

$$\begin{aligned}
\mathbf{SP}'_t: \quad V_t\left(x_{t-1}, y_{t-1}, \omega_t\right) = \min_{x_t} \max_{y_t} \quad & y_t^\top Q_t x_t + \mathcal{V}_{t+1}\left(x_t, y_t\right) \\
\text{subject to} \quad & A_t^{\omega_t} x_t + a^{\omega_t} \geq x_{t-1} \\
& y_t = B_t^{\omega_t} y_{t-1} + b^{\omega_t}.
\end{aligned}$$

It is in this form that it is apparent that the class of problems considered in this paper is a special case of those studied in Baucke et al. (2018). Note that $\mathbf{SP}'_t$ and $\mathbf{SP}_t$ yield the same optimal solution, since the feasible region for y_t is a singleton. In the next section we outline the method to derive saddle function cuts that provide lower bound approximations for the cost-to-go saddle function.

3. Saddle function lower bounds

Consider a saddle function $\mathcal{S}(x, y)$ with bounded subgradients, and suppose we have sampled it at points $(x_p, y_p) = p \in \mathcal{P}$. At each sampled point p, we acquire the true value of the function, θ_p, and a local subgradient vector of the function w.r.t. x, ϕ_p.

The remainder of this section provides a restatement of the derivation of saddle function lower bounds in Baucke et al. (2018) for our context. Let us define the following optimization problem:

$$\begin{aligned}
\mathcal{R}_{\mathcal{P}}(x, y) = \min_{\mu, \varphi} \quad & y^\top \mu + \varphi \\
\text{subject to} \quad & y_p^\top \mu + \varphi \geq \theta_p + \phi_p^\top (x - x_p), \quad \forall p \in \mathcal{P} \\
& ||\mu_t||_\infty \leq \alpha.
\end{aligned} \tag{6}$$

Lemma 3. *There exists a finite positive* α *such that* $\mathcal{R}_{\mathcal{P}}(x_q, y_q) = \mathcal{S}(x_q, y_q)$*, for all* $q \in \mathcal{P}$*, for any set of sample points* $\mathcal{P}$*.*

Proof. By substituting any sample point $q \in \mathcal{P}$ into (6), we know from the first constraint that $\mathcal{R}_{\mathcal{P}}(x_q, y_q) \geq \theta_q = \mathcal{S}(x_q, y_q)$. We will proceed to prove that there exists a μ^* yielding a candidate solution $(\mu, \varphi) = (\mu^*, \theta_q - y_q^\top \mu^*)$, with an objective function value of θ_q, which is feasible, and is therefore optimal.

Suppose we set μ^* to be a subgradient of $\mathcal{S}(x, y)$ with respect to y at (x_q, y_q). We know that $\mathcal{S}(x, y)$ has bounded subgradients, therefore there exists some finite α that will ensure that the second constraint of (6) is satisfied. Now since $\mathcal{S}(x, y)$ is concave with respect to y we know that:

$$(y_p - y_q)^\top \mu^* + \theta_q \geq \mathcal{S}(x_q, y_p), \quad \forall p \in \mathcal{P}. \tag{7}$$

Moreover, since $\mathcal{S}(x, y)$ is convex w.r.t. x we know that:

$$\mathcal{S}(x_q, y_p) \geq \theta_p + \phi_p^\top (x_q - x_p), \quad \forall p \in \mathcal{P}. \tag{8}$$

Equations (7) and (8) give

$$(y_p - y_q)^\top \mu^* + \theta_q \geq \theta_p + \phi_p^\top (x_q - x_p), \quad \forall p \in \mathcal{P},$$

which means that our candidate solution is feasible, and therefore $\mathcal{R}_{\mathcal{P}}(x_q, y_q) = \mathcal{S}(x_q, y_q)$. □

Theorem 1. *There exists a finite positive* α *such that* $\mathcal{R}_{\mathcal{P}}(x, y) \leq \mathcal{S}(x, y)$*, for any set of sample points* $\mathcal{P}$ *and all* (x, y)*.*

Proof. For the sake of contradiction, suppose there exists a $(\tilde{x}, \tilde{y})$ for which $\mathcal{R}_{\mathcal{P}}(\tilde{x}, \tilde{y}) > \mathcal{S}(\tilde{x}, \tilde{y})$. By defining $\mathcal{P}^* = \mathcal{P} \cup (\tilde{x}, \tilde{y})$, by Lemma 3, for some finite α, we have $\mathcal{R}_{\mathcal{P}^*}(\tilde{x}, \tilde{y}) = \mathcal{S}(\tilde{x}, \tilde{y})$. So, for the same α, $\mathcal{R}_{\mathcal{P}^*}(\tilde{x}, \tilde{y}) < \mathcal{R}_{\mathcal{P}}(\tilde{x}, \tilde{y})$, which is a contradiction since the feasible region of $R_{\mathcal{P}^*}$ is contained within $R_{\mathcal{P}}$ which can only increase the objective function value of $R_{\mathcal{P}^*}$. □

We note here that value functions $\mathcal{V}(x, y)$ defined in $\mathbf{SP}_t$, do indeed exhibit the property of bounded subgradients required for the bounding function $\mathcal{R}_{\mathcal{P}}(x, y)$ to be valid. This property arises from the following: $\mathbf{SP}_t$ is always feasible (admitting finitely bounded duals), cost coefficient matrices are bounded and finite, and finally that there are a finite number of stages.

4. Proposed algorithm and convergence

Baucke et al. (2018), provide an ϵ-convergence result for their algorithm which solves a more general class of problems, with minimax saddle function stage problems. This requires the definition of an upper bound function as well as a lower bound. In our setting, since y_t is exogenously determined we can forgo the computation of an upper bound function (and hence forgo deterministic convergence). Instead we prove the almost sure convergence of a different algorithm. Moreover, the formulation that we present here is more readily implementable in existing SDDP libraries (e.g. SDDP.jl (Dowson and Kapelevich, 2017)).

By Lemmas 1 and 2, the cost-to-go function $\mathcal{V}_{t+1}(x_t, y_t)$, defined in equation (1), is convex with respect to x_t, and concave with respect to y_t. Therefore, we can form an outer approximation using the saddle function representation from Section 3. In each iteration of the algorithm $k = 1, 2, \ldots$, we compute a set of feasible solutions $\{(x_t^k, y_t^k) : t = 1, \ldots, T-1\}$, and a set of cuts, one for each stage $t = 1, \ldots, T-1$. This gives rise to a sequence of approximate problems $\mathbf{AP}_t^k$, $k = 1, 2, \ldots$ for each stage, as defined below.

In each iteration k, for each stage $t = 1, \ldots, T$, we observe some random noise $\omega \in \Omega_t$ and solve the problem:

$\mathbf{AP}_t^k(x_{t-1}^k, y_{t-1}^k, \omega)$:
Set $y_t^k = B_t^\omega y_{t-1}^k + b^\omega$, and $c_t^{k\top} = y_t^{k\top} Q_t$, then solve:

$$
\begin{aligned}
\theta_t^k(x_{t-1}^k, y_{t-1}^k, \omega) = \min_{x_t, u_t, \varphi_t} \quad & c_t^{k\top} x_t + y_t^{k\top} \mu_t + \varphi_t \\
\text{subject to} \quad & A_t^\omega x_t + a^\omega \geq x_{t-1}^k \qquad [\pi_t^k] \\
& y_t^{j\top} \mu_t + \varphi_t \geq \alpha_t^j + \beta_t^{j\top} x_t \quad [\rho_t^{jk}], \quad j = 1, 2, \ldots, k-1 \\
& ||\mu_t||_\infty \leq \alpha.
\end{aligned}
$$

Note that for problem $\mathbf{AP}_1^k$ the inputs are fixed in advance (and invariant with respect to k) and recall that in the final stage T the cost-to-go is 0 so we can remove the μ_t and ϕ_t variables. Moreover, for a fixed ω, $\theta_t^k(x_{t-1}^k, y_{t-1}^k, \omega)$ is a saddle function.

Following Philpott and Guan (2008), we refer to a sequence of noise observations for stages $t = 1, \ldots, T-1$ as a scenario. The set of all possible scenarios is denoted by $\mathcal{S}$. Moreover, for a given scenario $s \in \mathcal{S}$ we denote the noise observation at the start of period t by ω_t^s. Algorithm 1, below, details the specific procedure to solve $\mathbf{SP}$, by solving $\mathbf{SP}_1(x_0, y_0, \omega_1)$.

In what follows, we show that this algorithm converges to an optimal solution in a finite number of iterations. First note that the key distinction between our

Algorithm 1: Scenario Enumeration

```
set k = 1
while not converged do
    set y_0^k = y_0, x_0^k = x_0, ω_1^k = ω_1
    for s ∈ S do
        /* Forward Pass                                              */
        for t = 1 : T − 1 do
            solve AP_t^k(x_{t−1}^k, y_{t−1}^k, ω_t^s)
            set x_t^k to the value of x_t in the optimal solution
        end
        /* Backward Pass                                             */
        for t = T − 1 : 1 do
            for ω ∈ Ω_{t+1} do
                solve AP_{t+1}^k(x_t^k, y_t^k, ω)
                set θ_{t+1}^{k,ω} to the optimal objective value θ_{t+1}^k
                set π_{t+1}^{k,ω} to the value of π_{t+1}^k in the optimal solution
            end
            set β_t^k = E_{ω∈Ω_{t+1}} [π_{t+1}^{k,ω}]
            set α_t^k = E_{ω∈Ω_{t+1}} [θ_{t+1}^{k,ω}] − β_t^{k⊤} x_t^k
            add the cut y_t^{k⊤} μ_t + φ_t ≥ α_t^k + β_t^{k⊤} x_t to the linear program for
             AP_t^k
        end
        increment k to k + 1
    end
end
```

problem and ones that can be solved using DOASA-type algorithms, as defined by Philpott and Guan (2008), is that due to the random noise affecting the objective function, the dual of $\mathbf{AP}_t^k$ (π_t^k, ρ_t^k) is not guaranteed to be feasible for all the outcomes of the random noise. Therefore, in order to construct valid (expected) cuts, every realization of the next stage's random noise must be sampled at each stage.

In order to prove that there is a finite number of possible cuts in each stage, we cannot *directly* invoke Lemma 1 from Philpott and Guan (2008), since we have a different stage problem. We present Lemma 4 below to show that the collection of distinct values of $(y_t^k, \alpha_t^k, \beta_t^k)$ is nevertheless provably finite.

Lemma 4. *For each $t = 2, 3, \ldots, T$, define the set*

$$\mathcal{G}_t^k = \left\{ \left(y_t^j, \alpha_t^j, \beta_t^j \right) : j = 1, 2, \ldots, k-1 \right\}.$$

Then for any sequence $\mathcal{G}_t^k, k = 1, 2, \ldots$ generated by Algorithm 1, there exists m_t such that for all k: $\left|\mathcal{G}_t^k\right| \leq m_t$. Furthermore, there exists k_t, so that if $k \geq k_t$ then $\mathcal{G}_t^k = \mathcal{G}_t^{k_t}$.

Proof. This lemma arises from the finiteness of the y_t vector, and given a fixed y_t, the finite number of extreme points of the feasible region in each stage problem. First, observe that by Assumption 1, there is a finite number of values that $y_t = \left\{ y_t^i : i = 1, 2, \ldots, I \right\}$ can take. Then, for a fixed value of y_t^i, we define the set

$$\mathcal{H}_t^{k,i} = \left\{ \left(y_t^i, \alpha_t^j, \beta_t^j \right) : j = 1, 2, \ldots, k-1 \right\}.$$

Since y_t^i is a constant, there exists an m_t^i such that for all k, $|\mathcal{H}_t^{k,i}| \leq m_t^i$. Furthermore, there is some k_t^i, so that if $k > k_t^i$, then $\mathcal{H}_t^{k,i} = \mathcal{H}_t^{k_t^i,i}$. Given that $\mathcal{G}_t^k$ can be expressed as

$$\mathcal{G}_t^k = \left\{ \mathcal{H}_t^{k,i} : i = 1, 2, \ldots, I \right\},$$

the remainder of the lemma can be proved using the same inductive technique as presented in Lemma 1 of Philpott and Guan (2008). □

Using the result of Lemma 4 (that there are a finite number of cuts possible at each stage) we will now show that Algorithm 1 is convergent.

Lemma 5. *Algorithm 1 converges after a finite number of iterations to an optimal solution, which is an optimal policy for* **SP***.*

Proof. By Lemma 4, for each $t \in \{2, \ldots, T\}$, there exists k_t, such that if $k > k_t$ then $\mathcal{G}_t^k = \mathcal{G}_t^{k_t}$. This means that there will be no further change in the cuts defining $\mathcal{V}_t\left(x_{t-1}, y_{t-1}\right)$. We define $\bar{k} = \max_t\{k_t\}$, to be the minimum number of iterations before all stages of the model no longer change.

Observe that since the stochastic process for y_t is discrete, there are a finite number of possible values for y_t for each $t \in \{2, \ldots, T\}$. Moreover, since $G_t^{k_t}$ is no longer changing for $k > k_t$, we arrive at a fixed sequence of controls $x_t(s)$ for each scenario[1]. Note that we also define the values of $y_t(s)$ in a similar way

[1]Since this is a dual dynamic programming problem the non-anticipatory requirements are met; that is, for two scenarios s_1 and s_2 with the same noise outcomes in stages $t \in \{1, \ldots, t^*\}$ we have $x_t(s_1) = x_t(s_2)$ for $t \in \{1, \ldots, t^*\}$.

(however, these are explicitly determined from the random noise outcomes for scenario s). We will now show that the controls are optimal for the problem $\mathbf{SP}_t$ for all stages and for each scenario.

First note that the optimal value function for $\mathbf{AP}_T^{\bar{k}}$ is the same as that for $\mathbf{SP}_T$, that is:

$$\mathbb{E}_{\omega_T \in \Omega_T}\left[\theta_T^{\bar{k}}\left(x_{T-1}(s), y_{T-1}(s), \omega_T\right)\right] = \mathcal{V}_T\left(x_{T-1}(s), y_{T-1}(s)\right), \ \forall s \in \mathcal{S}. \quad (9)$$

We now use induction to prove that at iteration $\bar{k}$ this is true for all $t \in \{1, \dots, T\}$. Suppose the analogous result to (9) is true for stage $t+1$, we will show that is it also true for stage t; that is:

$$\mathbb{E}_{\omega_t \in \Omega_t}\left[\theta_t^{\bar{k}}\left(x_{t-1}(s), y_{t-1}(s), \omega_t\right)\right] = \mathcal{V}_t\left(x_{t-1}(s), y_{t-1}(s)\right), \ \forall s \in \mathcal{S}. \quad (10)$$

We will prove this by contradiction; suppose for some scenario $\hat{s}$ we have

$$\theta_t^{\bar{k}}\left(x_{t-1}(\hat{s}), y_{t-1}(\hat{s}), \omega_t^{\hat{s}}\right) < V_t\left(x_{t-1}(\hat{s}), y_{t-1}(\hat{s}), \omega_t^{\hat{s}}\right).^2$$

However from Algorithm 1, we have

$$\begin{aligned} \theta_t^{\bar{k}}\left(x_{t-1}(\hat{s}), y_{t-1}(\hat{s}), \hat{\omega}\right) = \min_{x_t, u_t, \varphi_t} \quad & c_t^\top x_t + y_t^\top \mu_t + \varphi_t \\ \text{subject to} \quad & A_t^{\hat{\omega}} x_t + a^{\hat{\omega}} \geq x_{t-1}(\hat{s}) \\ & y_t^{j\top} \mu_t + \varphi_t \geq \alpha_t^j + \beta_t^{j\top} x_t \quad j = 1, 2, \dots, \bar{k}-1 \\ & ||\mu_t||_\infty \leq \alpha, \end{aligned}$$

where $y_t = B_t^{\hat{\omega}} y_{t-1}(\hat{s}) + b^{\hat{\omega}}$, $c_t^\top = y_t^\top Q_t$, and for notational convenience we set $\hat{\omega} = \omega_t^{\hat{s}}$. This has optimal solution $(x_t^*, u_t^*, \varphi_t^*)$, where $x_t^* = x_t(\hat{s})$.

If $y_t^\top \mu_t^* + \varphi_t^* < \mathcal{V}_{t+1}(x_t^*, y_t)$, from Lemma 3 we know that this means that there exists some random noise outcome, which we have not sampled in the backwards pass, since if we had sampled it we would have added a cut that would force $y_t^\top \mu_t + \varphi_t = \mathcal{V}_{t+1}(x_t^*, y_t)$. However, this contradicts the backward pass of Algorithm 1, which requires us to sample all random noise outcomes for the next stage to generate the expected cut. Given this contradiction and due to the enumeration of scenarios in the forward pass of Algorithm 1, equation (10) must hold. Therefore, after a finite number of iterations $\bar{k}$, Algorithm 1 converges to the optimal solution for $\mathbf{SP}$. $\square$

[2]By Theorem 1 we are adding cuts that are guaranteed to not exceed the cost-to-go, so $\theta_t^{\bar{k}}$ cannot exceed V_t.

Although we have shown that Algorithm 1 converges, it is computationally intractable, since it enumerates the scenarios. We therefore construct a sampling-based algorithm (Algorithm 2). Instead of sampling all scenarios $s \in \mathcal{S}$ in an iteration before checking for convergence, Algorithm 2 randomly and independently samples a single scenario. In Theorem 2 we prove almost sure convergence of Algorithm 2 in a finite number of iterations, but first we present a short corollary.

Corollary 1. *Suppose, in each stage* $t \in \{1, \ldots, T-1\}$, *there exists in* $\mathbf{AP}_t^1$ *some initial set of cuts that form a valid lower bound for the true cost-to-go function* $\mathcal{V}_{t+1}(x_t, y_t)$. *Then Lemma 5 still holds if we begin Algorithm 1 from this point.*

Proof. Introducing additional valid cuts to $\mathbf{AP}_t^1$ has no bearing on the application of Theorem 1 and Lemma 4 in the convergence proof of Lemma 5. Although the set of cuts produced will be different, there are still a finite number. □

Theorem 2. *Algorithm 2 converges with probability 1 to an optimal solution to* **SP** *in a finite number of iterations.*

Proof. Due to the forward pass sampling method given in Algorithm 2, without the stopping criteria, we know, by the second Borel-Cantelli lemma (Grimmett and Stirzaker, 1992), that after some iterations we would almost surely realize a sequence of iterations that is identical to those traversed using Algorithm 1. This gives us an initial set of cuts before the iterations of Algorithm 1, and thus from Corollary 1, we know that Algorithm 2 will converge to the optimal solution to **SP** with probability 1. □

5. Results

We implement the hydro-bidding problem with price uncertainty example from Chapter 6.1 of Wahid (2017). The example has two reservoirs in a cascade over a planning horizon of 12 periods (i.e. $T = 12$). On the outflow of each reservoir is a turbine followed by a river that leads to the next reservoir in the cascade. The last reservoir discharges into the sea. The goal of the agent is to choose the quantity of water to flow through each turbine (and thereby, produce electricity), as well as the quantity of water to spill over the top of each reservoir, in order to maximize the revenue gathered from selling the electricity generated on the spot market. In each stage t, we denote the combination of spills and flows as u_t. We denote the feasibility constraints on the action of the agent (i.e. maximum and minimum flow limits) as $u_t \in \mathcal{U}$. Based on the action u_t chosen by the agent, the state of the system x_t (the quantity of water in each of the reservoirs at the start of stage t) will transition according to the linear function $x_t = \mathcal{F}(x_{t-1}, u_t)$. In addition, the agent earns an immediate reward by selling $\mathcal{G}(u_t)$ units of power at the current spot-price y_t. They may also incur a penalty of $\mathcal{C}(u_t)$ if they exceed some operating limits of the reservoir (i.e. a cost of spillage). Unless explicitly stated, all parameters are identical to the description given in Wahid (2017).

The spot-price process y_t was modelled as the auto-regressive lag 1 process: $y_t = 0.5y_{t-1} + 0.5b(t) + \omega_t$, where $\Omega_t = \{-1.65, -1.03, -0.67, -0.38, -0.12, 0.12, 0.38, 0.67, 1.03, 1.65\}$. Each of the realizations ω_t in Ω_t is sampled with uniform probability. This hydro-bidding model can be described by the stage problem:

$$
\begin{aligned}
\mathbf{SP}_t: \; V_t(x_{t-1}, y_{t-1}, \omega_t) = \min_{x_t, y_t} \quad & \mathcal{C}(u_t) - y_t^k \mathcal{G}(u_t) + \mathbb{E}_{\omega_{t+1} \in \Omega_{t+1}}\left[V_{t+1}(x_t, y_t, \omega_{t+1})\right] \\
\text{subject to} \quad & x_t = \mathcal{F}(x_{t-1}, u_t) \\
& u_t \in \mathcal{U} \\
& y_t^k = 0.5y_{t-1}^k + 0.5b(t) + \omega_t.
\end{aligned}
$$

This model was implemented in the SDDP.jl package (Dowson and Kapelevich, 2017) in the Julia language (Bezanson et al., 2017). The model was solved using Algorithm 2 for 2000 iterations. In addition, we solved the same model using the *stochastic dynamic programming* (SDP) algorithm (implemented in the DynamicProgramming.jl package (Dowson, 2017)). The lower bound for the problem using Algorithm 2 converges to the first-stage objective value (calculated using the SDP algorithm) of $-\$24{,}855$ (see Figure 1).

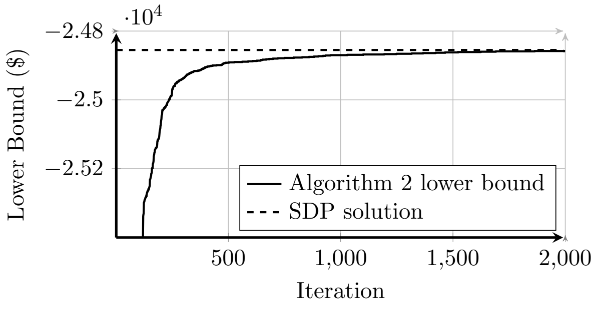

Figure 1: Convergence of the lower bound against the number of iterations.

In this simple model, the SDP approach is faster than Algorithm 2 (although it depends on the level of discretization of the state and action spaces). However, it is well known that SDP algorithm is limited by the "curse of dimensionality". To demonstrate the ability of Algorithm 2 to escape the curse, we extend the model to a cascade of five reservoirs, and the price process to an auto-regressive process with lag 2: $y_t = 0.5y_{t-1} + 0.5(y_{t-1} - y_{t-2}) + 0.5b(t) + \omega_t$.

There are now seven state-dimensions in the model (the five reservoir levels, plus the two prices y_{t-1} and y_{t-2}), and five action-dimensions (the quantity of water to release from each reservoir). This problem is too large to solve in a reasonable time using the SDP algorithm. However, the enlarged problem was solved using Algorithm 2 for 5000 iterations to form an approximately optimal policy. Then, a Monte-Carlo simulation was conducted with 1000 replications using the policy. This took approximately 2000 seconds using a single core of a Windows 7 machine with an Intel i7-4770 CPU and 16GB of memory. In Figure 2, we visualize the Monte-Carlo simulation. In each of the subplots, we plot as shaded bands in order of increasing darkness, the 0–100, 10–90, and 25–75 percentiles of the distribution of the plotted variable. The solid line corresponds to the 50$^{\text{th}}$ percentile. There are also two individual replications plotted: a high-price realization (thick dashed line) and low-price realization (thick dotted line).

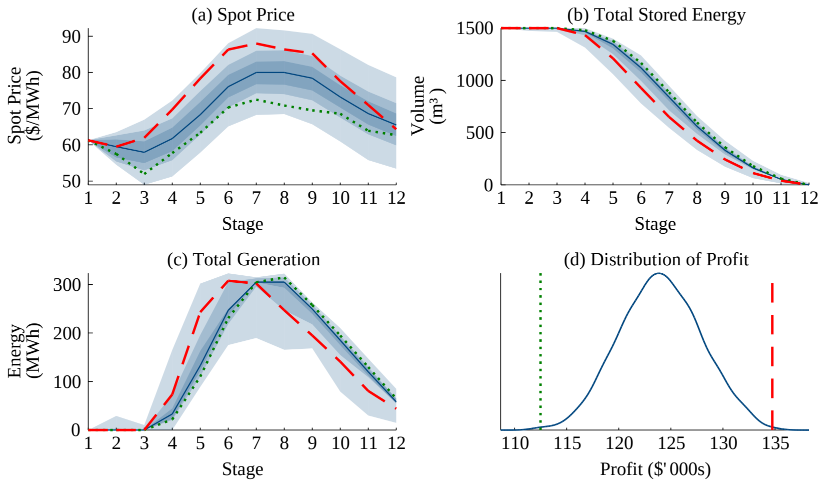

Figure 2: Monte-Carlo simulation using the optimal policy.

Over the stages 1–7, the spot-price increases (Figure 2a). Therefore, optimal policy is to conserve water in anticipation of a future higher profit. In Figure 2b, we plot the total stored energy in the system. ($1m^3$ in Reservoir 1 is worth five units as it can be used by each of the five turbines in the cascade. In contrast, $1m^3$ in Reservoir 5 is only worth one unit as it can only be used once before it flows out of the cascade.) In the low-price scenario (thick dotted line), more energy is stored in the system at a given point in time in the expectation that the spot-price will revert upwards to the mean, whereas in the high-price scenario (thick dashed line), more energy is generated (Figure 2c) in the early stages (i.e. stages 1-5) in anticipation that the futures prices will revert downwards to the mean. This leads to the distribution in profit shown in Figure 2d.

6. Conclusions

In this paper we have presented an algorithm which converges almost surely in a finite number of iteration when solving linear multistage stochastic programming problems with stagewise dependent noise in the objective function. Moreover, this algorithm has been implemented within the SDDP.jl Julia library (Dowson and Kapelevich, 2017).

This method enables the modelling of stochastic price processes, such as pure auto-regressive (AR) models or ones incorporating a moving average (ARMA), within the SDDP framework. These types of problems could previously only be approximated using Markov chains (Gjelsvik et al., 1999), or binary expansion techniques (Zou et al., 2016).

Furthermore, note that the algorithm is not limited to price processes, and is general to any objective function state variable, so there are many applications for this method. The key requirement is that these state variables evolves exogenously to the control decisions and the values of the other state variables.

Finally, in this paper we have elected to avoid a detailed discussion of the practical implementation of the algorithm. For example, in the initial iterations, it is necessary to bound the value of $y_t^k \mu_t + \varphi_t$ by choosing α and introducing a lower bound on the φ_t variables. The choice of an appropriate bounds can affect the speed of convergence. We provide Julia code to implement Algorithm 2 as supplementary materials at `https://github.com/odow/SDDP.jl`.

Acknowledgements

We would like to acknowledge Andy Philpott and Andrew Mason for useful discussions on this work.

References

Baucke, R., Downward, A., Zakeri, G., 2018. A deterministic algorithm for solving multistage stochastic minimax dynamic programmes. Optimization Online.
URL `http://www.optimization-online.org/DB_HTML/2018/02/6449.html`

Bezanson, J., Edelman, A., Karpinski, S., Shah, V. B., 2017. Julia: A Fresh Approach to Numerical Computing. SIAM Review 59 (1), 65–98.

Dowson, O., 2017. DynamicProgramming.jl: a Julia package for Stochastic Dynamic Programming. [Online; accessed 2017-10-26].
URL `https://github.com/odow/DynamicProgramming.jl`

Dowson, O., Kapelevich, L., 2017. SDDP.jl: a Julia package for Stochastic Dual Dynamic Programming. Optimization Online.
URL `http://www.optimization-online.org/DB_HTML/2017/12/6388.html`

Gjelsvik, A., Belsnes, M., Haugstad, A., 1999. An algorithm for stochastic medium term hydro thermal scheduling under spot price uncertainty. In: PSCC: 13th Power Systems Computation Conference : Proceedings. Executive Board of the 13th Power Systems Computation Conference, 1999, Trondheim, p. 1328.

Gjelsvik, A., Mo, B., Haugstad, A., 2010. Long- and Medium-term Operations Planning and Stochastic Modelling. In: Pardalos, P. M., Rebennack, S., Pereira, M. V. F., Iliadis, N. A. (Eds.), Handbook of Power Systems I. Energy Systems. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 33–55, dOI: 10.1007/978-3-642-02493-1.

Grimmett, G., Stirzaker, D., 1992. Probability and Random Processes. Oxford University Press, Oxford.

Pereira, M. V. F., Pinto, L. M. V. G., 1991. Multi-stage stochastic optimization applied to energy planning. Mathematical Programming 52 (1-3), 359–375.

Philpott, A. B., De Matos, V. L., 2012. Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion. European Journal of Operational Research 218 (2), 470–483. URL `http://dx.doi.org/10.1016/j.ejor.2011.10.056`

Philpott, A. B., Guan, Z., 2008. On the convergence of stochastic dual dynamic programming and related methods. Operations Research Letters 36 (4), 450–455.

Rebennack, S., Mar. 2016. Combining sampling-based and scenario-based nested Benders decomposition methods: application to stochastic dual dynamic programming. Mathematical Programming 156 (1-2), 343–389.

Wahid, F., 2017. River optimization: short-term hydro-bidding under uncertainty. PhD thesis, University of Auckland, Auckland, New Zealand.

Zou, J., Ahmed, S., Sun, X. A., 2016. Stochastic Dual Dynamic Integer Programming. Optimization Online.
URL `http://www.optimization-online.org/DB_HTML/2016/05/5436.pdf`